

**Lay-out, Structuur en Inhoud voor verslagen Elektrotechniek**

*Versie mei 2017; auteurs Michiel Scager, Petra Verbeek*

Dit *format Verslag* is een werkdocument op 'metaniveau' voor direct gebruik door de student. Het geeft handvatten voor en inzicht in de structuur, de inhoudelijke aspecten en de lay-out van verslagen. Het *format Verslag* sluit aan op het boek Rapportagetechniek van Elling e.a.

Dit format wordt gebruikt als

* **Richtlijn**, waar het de *structuur, volgorde* (van hoofdstukken) *en inhoud* betreft;
* **Standaard**, waar het gaat om *vormaspecten* (font, regelafstand, hoofdstuknummering, footers, paginanummering) en *standaardelementen* (verplicht: titelpagina, inhoudsopgave, samenvatting, inleiding, hoofdstukken. Bij de meeste verslagen horen tevens conclusies en aanbevelingen en bijlagen. Zijn gegevens ontleend aan internet of literatuur, dan is een bronvermelding verplicht).

Het wordt van de student verwacht, dat hij dit format gebruikt en zodanig aanpast, dat het de context en de inhoud optimaal dient. Daartoe moeten onder meer de begeleidende teksten tussen [vierkante haken] gewist en/of vervangen worden door eigen tekst. Tevens dienen uiteraard documentnaam, titel en voetteksten aangepast te worden.

[Voorblad. Voor een verslag kleiner dan 20 of 25 A4 is een voorblad niet noodzakelijk: voorblad en titelpagina vallen dan samen. Op het voorbladkarakteriseert de *titel* kort en krachtig de inhoud. Gebruik geen afkortingen in de titel. Bij een omvangrijk rapport (bv. afstudeerverslag) bestaat de titel uit een *hoofdtitel* met een *informatieve ondertitel* om de inhoud te specificeren.

Onder de titel wordt een *illustratie* (bv. van het eindproduct) en/of bedrijfslogo geplaatst.

Op het voorblad dienen tevens minimaal *klas, projectgroep, naam, studentnummer, cursus, cursuscode, cursusonderdeel* en *datum* aangegeven te worden, bv. zoals in onderstaande tabel. Print het voorblad in kleur, de titelpagina (hierna) kan in zwart-wit afgedrukt worden. Het voorblad is van stevig papier.]



**Lay-out, Structuur en Inhoud voor verslagen Elektrotechniek**

[De *titelpagina* komt altijd op een nieuw blad terecht, dus niet op de achterzijde van het voorblad.

De *titelpagina* is doorgaans een kopie van het voorblad, maar nu met meer informatie. Deze informatie heeft de lezer nodig om een bronvermelding van het verslag te maken of om het verslag op te vragen. Denk aan: school, faculteit, instituut, opleiding / projectgroep, naam van het project/opdracht, studentnaam, -nummer, telefoonnummer / cursusonderdeel, cursuscode, namen van de docenten/tutoren/examinatoren / plaats en datum van uitgave / versiebeheer.

De informatie van het voorblad wordt één op één overgenomen (zelfde lettertype, -grootte, positie). De illustratie van het voorblad valt op de titelpagina weg.]

[Verplicht op te nemen tekst]

Het bestuur van de Stichting Hogeschool Utrecht te Utrecht aanvaardt geen enkele aansprakelijkheid voor schade voortvloeiende uit het gebruik van enig gegeven, hulpmiddel, werkwijze of procedure in dit verslag beschreven. Vermenigvuldiging zonder toestemming van de auteur(s), de school of het bedrijf (indien van toepassing) is niet toegestaan.

**Inhoudsopgave**

[De *inhoudsopgave* is niet alleen een instrument voor de lezer om informatie in het verslag te vinden, maar geeft ook de opbouw van het verslag weer. Daartoe is het belangrijk dat de koptitels van hoofdstukken, paragrafen en subparagrafen informatief zijn.

Let op de volgende punten:

* Onderdelen decimaal nummeren: hoofdstuk 1, hoofdstuk 2 etc.; paragraafkopjes 1.1 en 1.2 etc. Gebruik geen dubbele punten.
* Houd de inhoudsopgave compact: een 'diepte' van 2 niveaus is doorgaans voldoende (wel: 1.1, 1.2 – liever niet: 1.1.1)
* De onderdelen inhoudsopgave, samenvatting, voorwoord, inleiding, lijst van afkortingen, bronnen en bijlagen krijgen géén hoofdstuknummering.
* Titelpagina, voorwoord en inhoudsopgave maken geen deel uit van de inhoudsopgave.
* Eventuele bijlagen komen achteraan en moeten voorzien zijn van een titel en van een eigen nummer als er meerdere zijn (A, B, … of I, II …). Bv. Bijlage A Tabellen
* De paginanummering loopt (mits mogelijk) in de bijlagen door.
* Controleer vóór inleveren altijd of de inhoudsopgave up-to-date is.

Onderstaande inhoudsopgave is automatisch opgesteld.]

[Introduction 6](#_Toc62756930)

[1 Assignment 8](#_Toc62756931)

[1.1 Assignment description 8](#_Toc62756932)

[1.2 Goals and deliverables 9](#_Toc62756933)

[1.3 Criteria 10](#_Toc62756934)

[2 Realization 11](#_Toc62756935)

[2.1 DFT 11](#_Toc62756936)

[2.2 FSB 17](#_Toc62756937)

[2.3 Visualization 20](#_Toc62756938)

[3 Resultaten 22](#_Toc62756939)

[3.1 Eindproduct 22](#_Toc62756940)

[3.2 Antwoord op de deelvragen 22](#_Toc62756941)

[4 Conclusie en aanbevelingen 23](#_Toc62756942)

[4.1 Antwoord op de hoofdvraag 23](#_Toc62756943)

[4.2 Aanbevelingen 23](#_Toc62756944)

[Afkortingen en begrippen 24](#_Toc62756945)

[Lijst van Illustraties 24](#_Toc62756946)

[Gebruikte afkortingen en symbolen 24](#_Toc62756947)

[Verklarende woordenlijst 24](#_Toc62756948)

[Index 24](#_Toc62756949)

[Bronnen 25](#_Toc62756950)

[Bijlagen 26](#_Toc62756951)

[Bijlage I: Aanwijzingen voor de schrijver 27](#_Toc62756952)

[Bijlage II: De APA-standaard voor bronvermelding 30](#_Toc62756953)

[Bijlage III: Plagiaat 31](#_Toc62756954)

[Bijlage IV: Dit format en SCRUM 32](#_Toc62756955)

[Let op. De voettekst (titel, paginanummering, aantal pagina’s) begint volgend op de pagina met de inhoudsopgave. Daarom is vanaf hier een Word-sectie toegevoegd (druk ctrl-shift-8 om de speciale tekens te zien).]

# Introduction

For the 5LIU0 course, the students were allowed to define their own project related to signal processing or feedback control. During my previous education both subjects were touched upon, control systems much more than signals. The idea was to combine a signals related project with a control related project.

During my previous education I struggled with understanding the practical side of the signals course. For a group project, my partners wrote their implementation of a Discrete Fourier Analysis, while I wrote a VGA controller and USB receiver for an FPGA for the game we were developing. Needless to say, I wanted to prove to myself I could do it.

As for the control prt, simulations is what we did during those courses. Once we wrote a PID controller, but not from the ground up. I would like to write my own controller just to see if I could do it and what challenges I would face.

This report details my journey through this project.

Chapter walkthrough…

[In de *eftding* wordt beschreven waar het verslag over gaat én hoe het opgebouwd is. De eftding is parafraserend, verhalend, bedoeld. Anders dan de samenvatting bevat de eftding dan ook *géén technisch-inhoudelijke tekst*, zoals de eft aan het product (die worden namelijk apart vermeld). De tekst in de eftding mag in principe niet in dezelfde vorm later in het verslag terugkomen.

Na het lezen van de eftding moet de lezer op de hoogte zijn van de *aanleiding*, de *doelstelling* en de *structuur* van het verslag. De volgende onderdelen moeten terugkomen (in drie eftd’s of bij een omvangrijk verslag drie paragrafen):

* *Aanleiding.* Beschrijf kort waarom de tekst voor de lezers belangrijk/interessant is en waarom de opdracht is uitgevoerd. Schets heel kort de situatie en plaats het onderwerp in een groter kader. Zorg dat de lezers begrijpen dat het belangrijk is dat het eftdi wordt opgelost of de vraag van de opdrachtgever wordt beantwoord.
* *Doelstelling.* Beschrijf in lopende zinnen waar het om gaat aan de hand van de hoofdvraag die in het verslag wordt beantwoord en de wijze waarop geprobeerd een antwoord op deze vraag te vinden. Bij officiële documenten (bv. Afstudeerverslag): beschrijf waar jij je mee bezighoudt, wat jij binnen de opdracht gaat doen.
* *Structuurbeschrijving/leeswijzer*. Schets de grote lijn van het verslag. Geef in lopende zinnen aan hoe het verslag is opgebouwd en welke relatie de hoofdstukken (en paragrafen) tot elkaar hebben. Een herhaling of samenvatting van de inhoudsopgave volstaat niet.

De eftding eft geenhoofdstuknummering.]

# Assignment

This chapter details the background knowledge and other base ideas about this project.

This chapter details how and why the project changed after the project proposal. The goals and deliverables are updated. Success criteria are also given.

## Assignment description

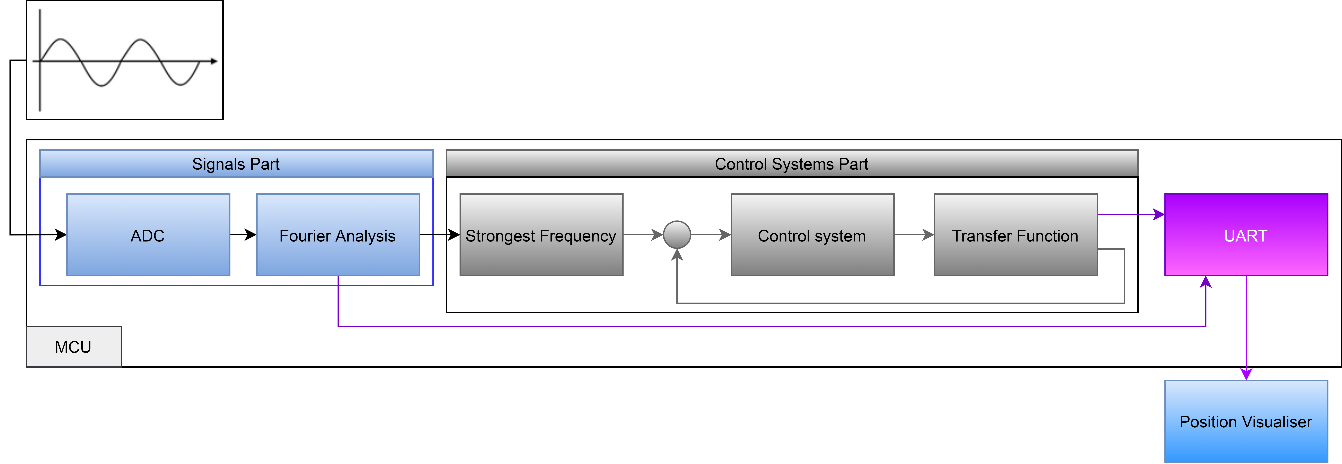


Figure 1 Block Diagram of the Project Proposal

The goal for this project was to combine Signals and Control systems projects into one project, see Figure 1. In my previous education (HBO Embedded Systems) both concepts were touched upon separately (control systems much more than signals).

For the signals part the first two proposed signals related projects in from the 5LIU0 Study Guide were implemented:

* A selected, real-time, signal processing algorithm, for instance filtering, or feature extraction, implemented in software on a pc, or on a microprocessor board.
* Signal visualization. Extract and visualize relevant features from an input signal. For instance, related to energy or spectral content of the signal.

Originally a microcontroller would sample an analog signal using its build-in ADC, after which a Fourier analysis would be done to find the strongest frequency present in the sampled signal.



Figure 2 Block Diagram of the robot arm and the transfer function

This strongest frequency, a number, would be used as the reference to the control part of this project. Either a PID or Full-State Feedback controller would be written in C for the microcontroller. The plant is a transfer function of an assignment from my previous education, Figure 2.

We worked with a simulated robot arm to determine if a PID or Full-State Feedback controller would be best for that case. The robot arm simulation model is only used for its transfer function. The position of the robot arm is determined by the input frequency.

Using the UART of the microcontroller information would be sent to the pc to visualize the Fourier Analysis and the control system output.

Shortly after the start of the project, it was decided to not program an embedded system. The function generator used for generating the sinusoidal signal either had bad spectral purity or there was something wrong with the microcontroller. Either way, it could not be determined as due to corona virus measures the campus (HU) was closed. I had no access to the labs to determine what the problem was. It was decided to write all the programs that would run on the microcontroller on Windows in C. If it needed porting to a microcontroller it could easily be done.

## Goals and deliverables

Table 1 shows the new goals of this project. The project is broken up into manageable pieces, shown in Figure 3 .

Table 1

|  |  |
| --- | --- |
| Goals | |
| 1. Writing a Fourier Analysis in C. 2. Writing my own PID or Full-State Feedback controller in C. | |
| Deliverables | |
| Must | **Could** |
| 1. A demonstration of the Signals part with the visualizer 2. The software with detailed documentation | 1. A demonstration of the Control part with the visualizer 2. A demo of the Signal and Control part working together with the visualizer |

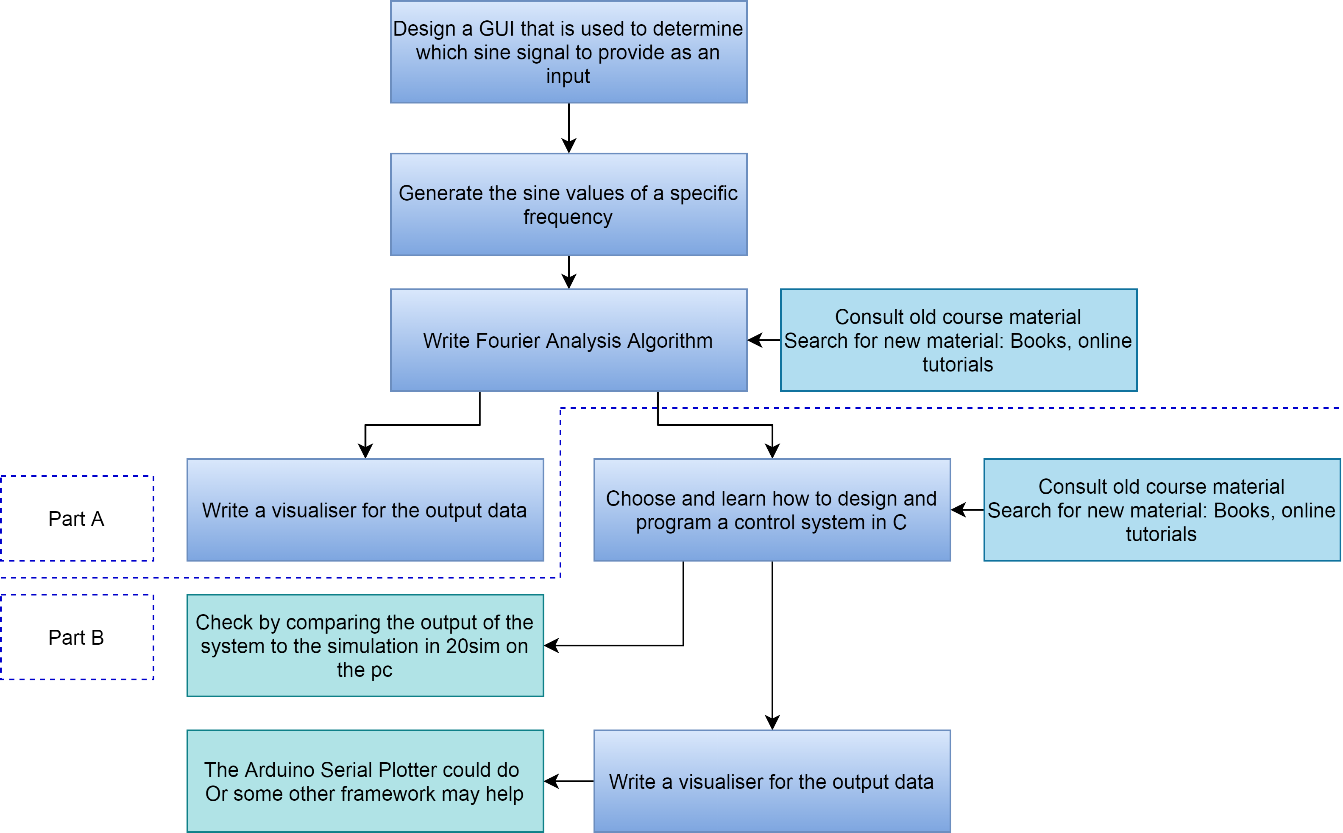


Figure 3 Flow chart project progression

## Criteria

~~No criteria were set at the beginning of the project. My personal goal was to see how far I could get. Because the output of the various written program could be checked against known to be good values (expected values, professional simulations) anything that could not be explained a s a rounding error would be pursued until it became “acceptable”.~~

rewrite

# Realization

## DFT

For my previous education, we used “The Scientist and Engineer’s Guide to Digital Signal Processing” book to learn about DSP (Smith, 1999). In chapter 8, there a Discrete Fourier Analysis program is given, written in BASIC. Code Snippet 1 gives the translated version in C written by me.

The book does a great job explaining how the DFT works. Simply put: it describes how an individual sample in the frequency domain is affected by all of the samples in the time domain (Smith, 1999).

This paragraph details the creation and verification of the DFT program written in C.

### Baseline

Before writing the C code, Excel and an online tool were prepared to set expectations for the final C code. Matlab would also have used, but at the time the written Matlab code did not give satisfactory output.

Excel was used for:

1. Creating the sine values for the online tool
2. Performing it’s build-in FFT function on the same sine values

A sine wave of 100Hz with a sample frequency of 1kHz was created in Excel. The output of Excel is given in Figure 4 and the output of the online tool (using the same sine values) is given in Figure 5. Note that Excel cuts of at half the sampling frequency while to online tool goes up to the sampling frequency. This is intentional, as the frequencies above the sampling frequency are not used.

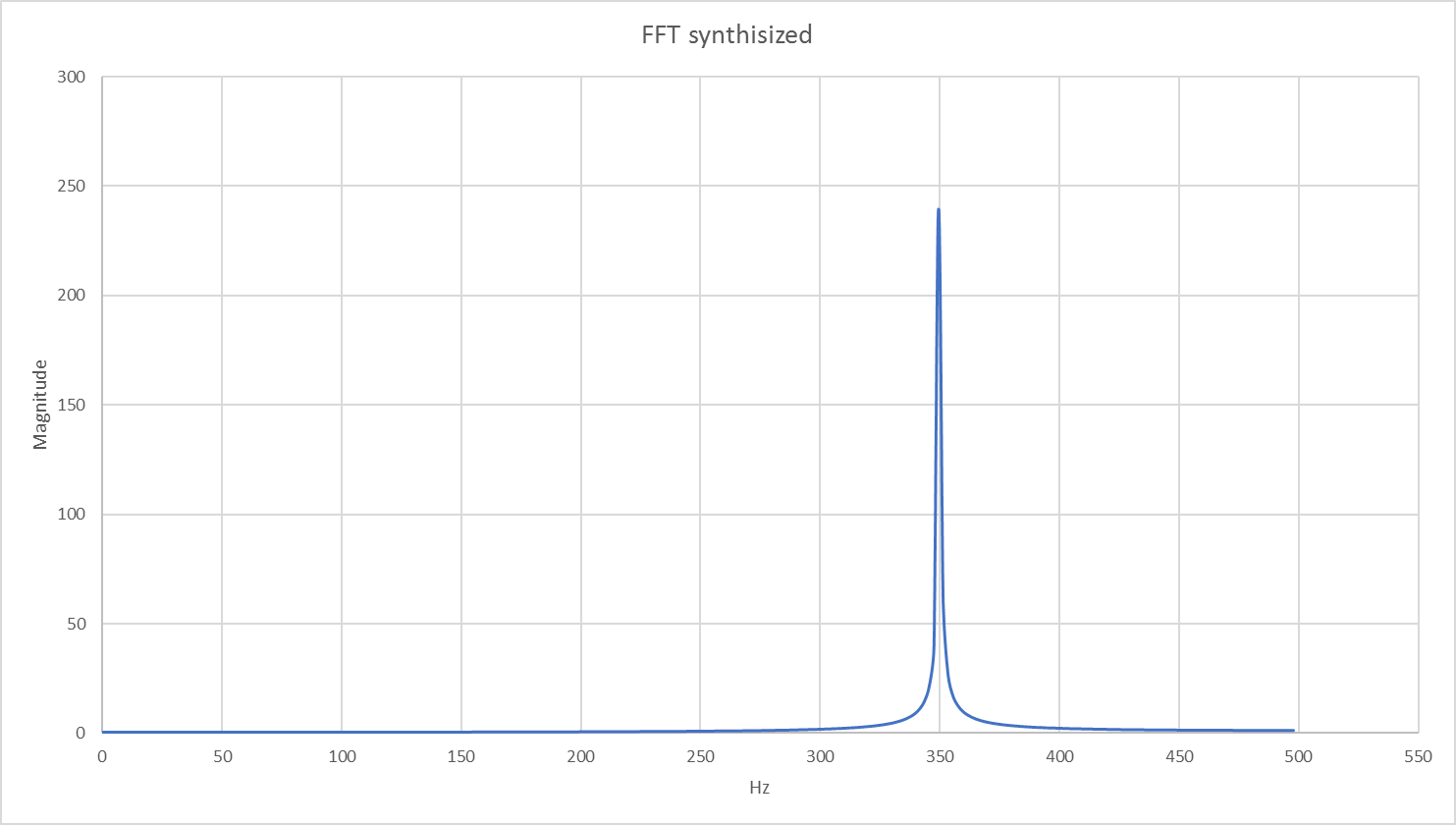


Figure 4 Excel DFT. Fsample = 1000Hz, Fsignal = 300Hz, 256 DFT values

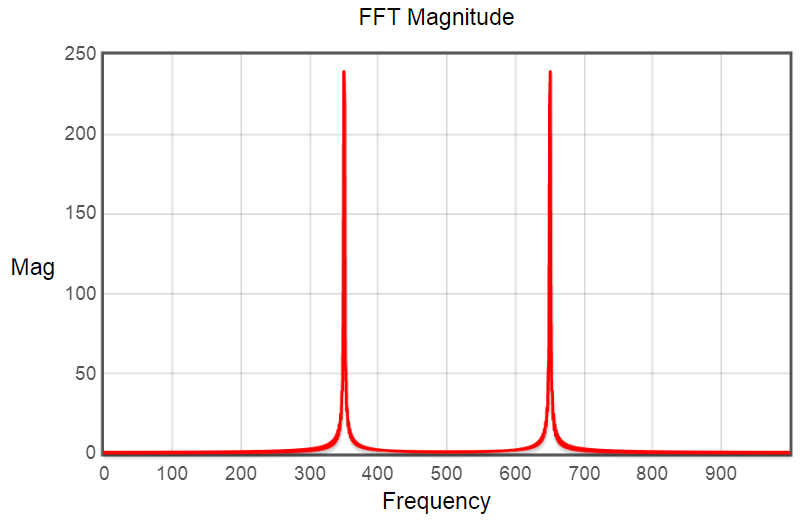


Figure 5 FFT Magnitude plot from the online tool (Ahmed, 2013)

### C code

Figure 6 gives the high level overview of the steps taken by the DFT program. Code Snippet 1 gives the lines of code that actually calculate the DFT

A high level overview of the written program is given in Figure 6. The program generates its own sine values. The reasoning was, if this would be run on an embedded system one would only need to send some parameters instead of a lot of sine values. The sine values are stored in a text file for later use.

The actual DFT code is given in Code Snippet 1. Two arrays are used to store the real and imaginary outputs of the DFT. These are stored in their own text file. Excel would be used to plot the output.

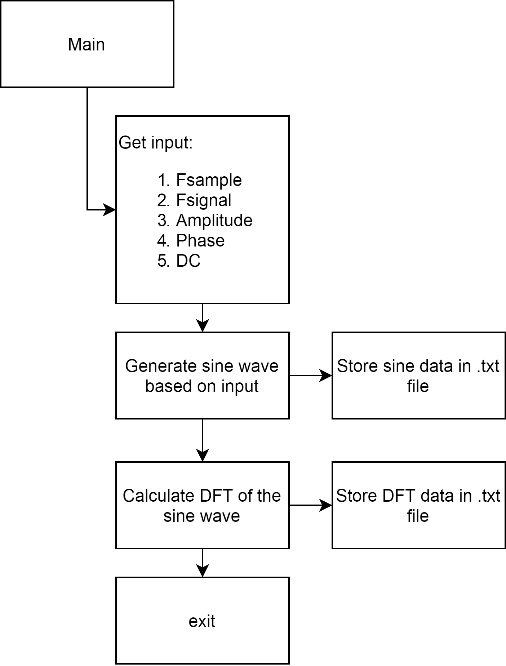
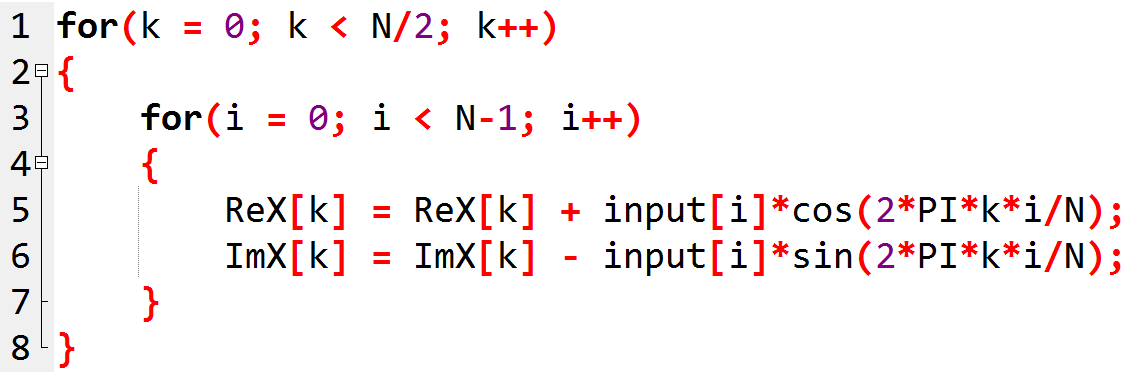


Figure 6 Flow chart dft.c



Code Snippet 1 DFT in C

### Verification

The output of the C code will be compared against Excel, the online tool and Matlab. After writing the dft program I managed to put together Matlab code to plot the fft of given sine values.

A sinewave of 1600Hz with a sampling frequency of 5000Hz was generated.

Figure 7 gives the output of the dft.c program. Figure 8 gives the Excel output, Figure 9 the online tool and Figure 10 gives the Malab output. The matlab code is shown in Code Snippet 2.



Code Snippet 2 Matlab code

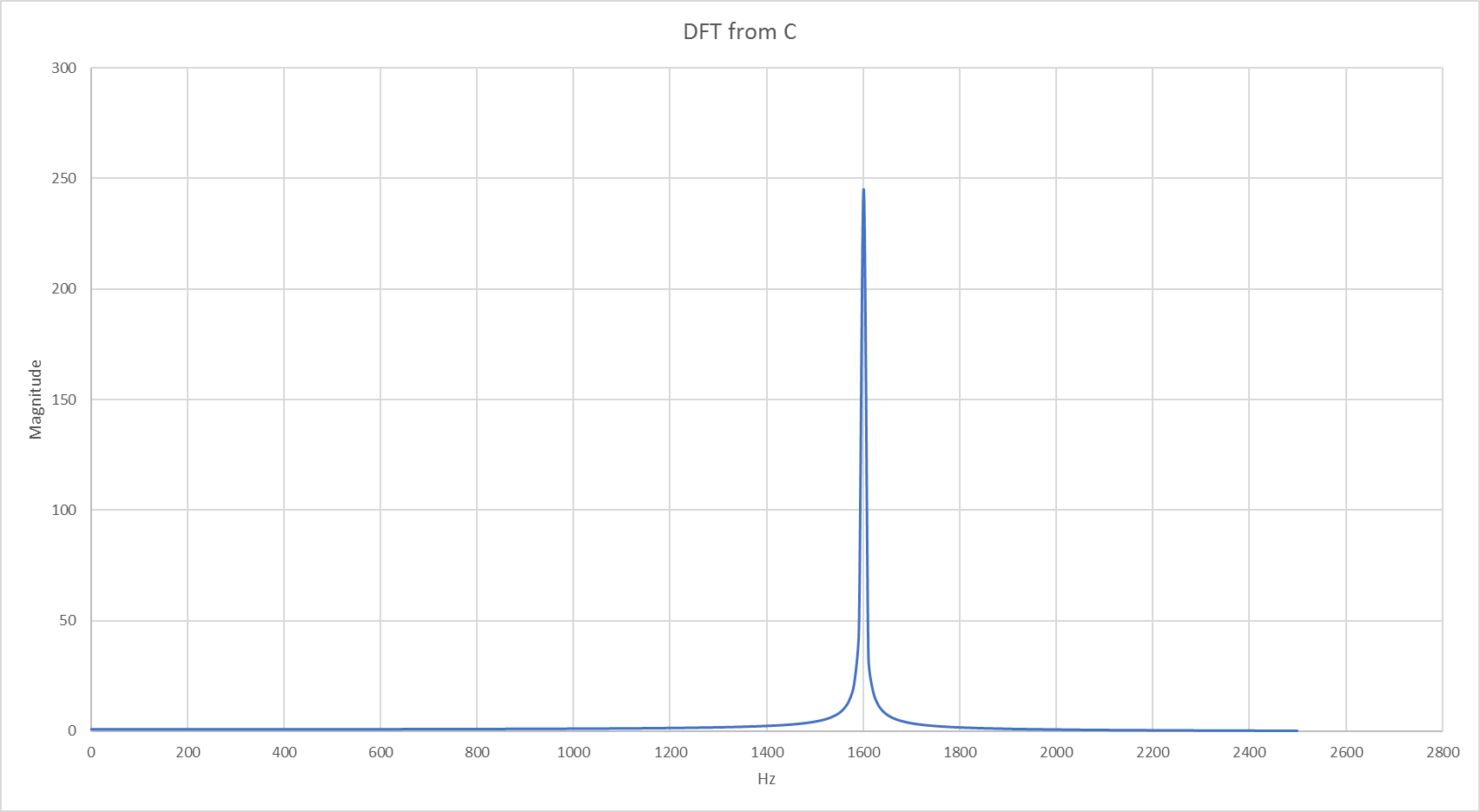


Figure 7 DFT calculated by C code

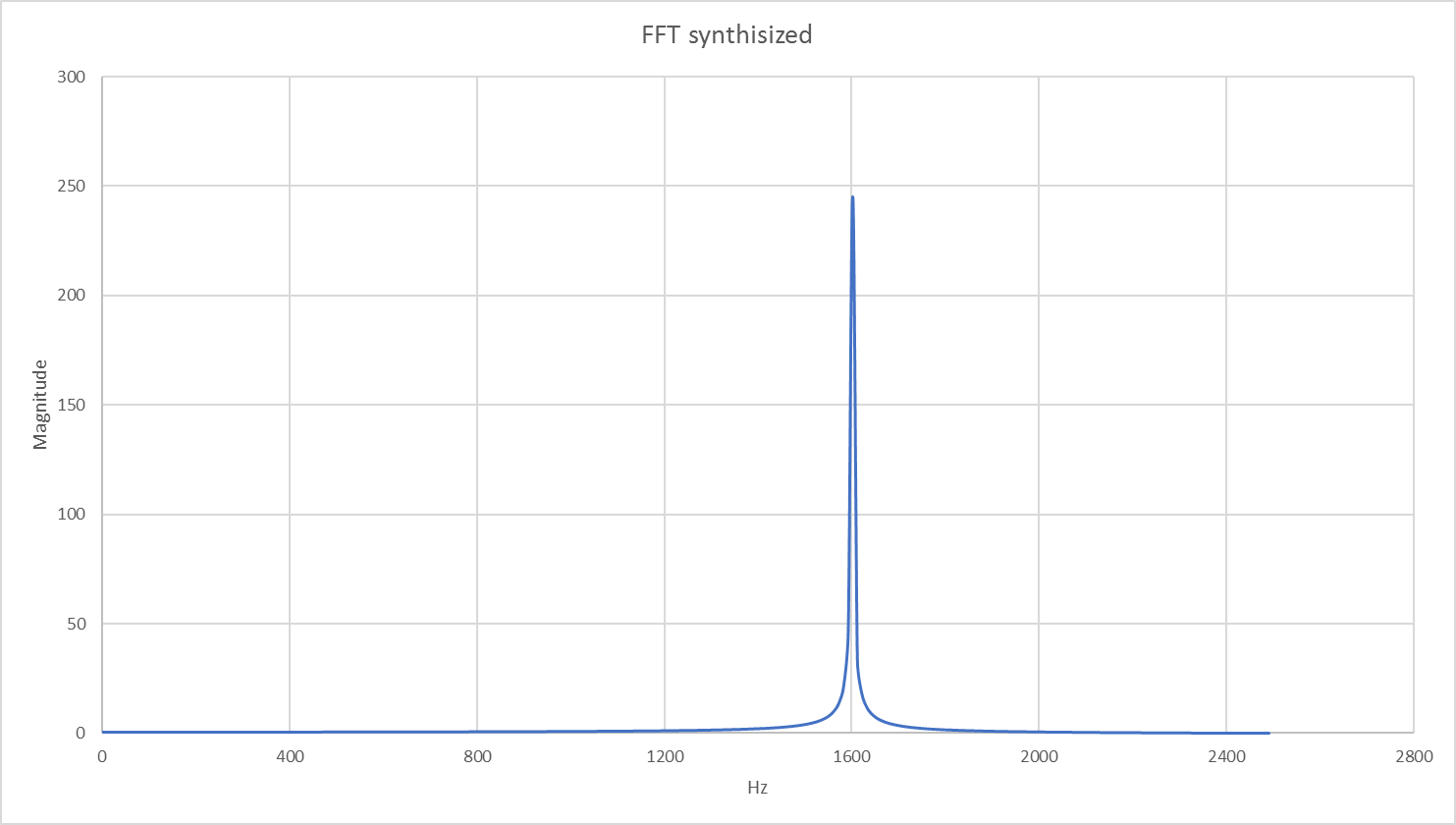


Figure 8 Excel fft

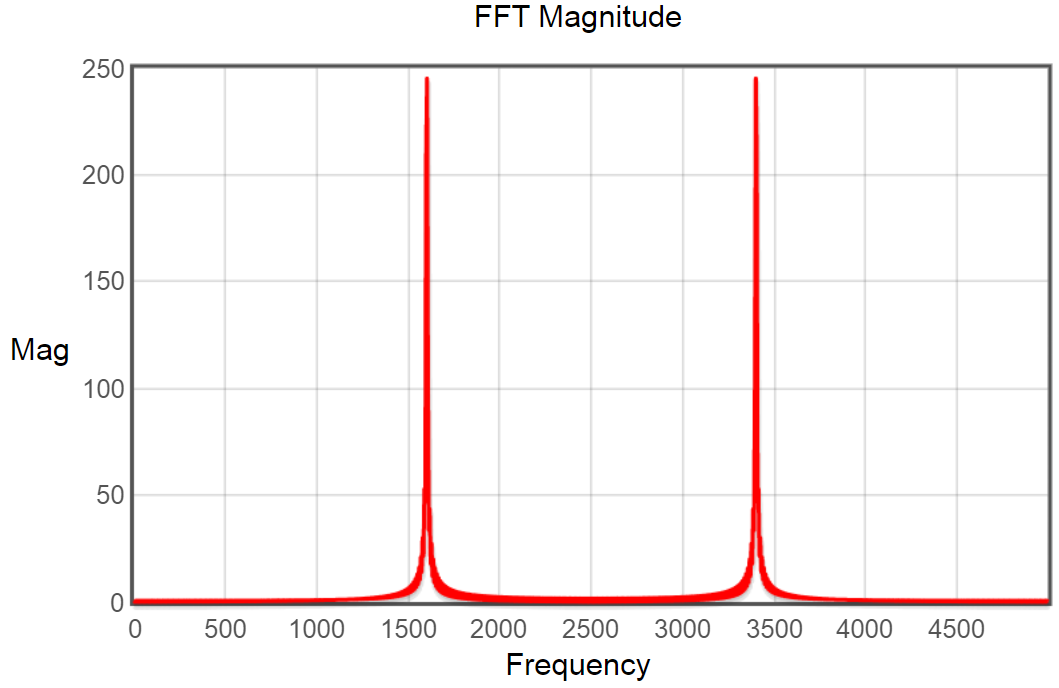


Figure 9 Online tool output

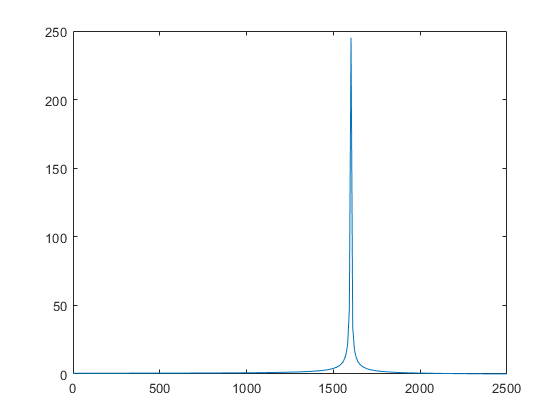


Figure 10 Matlab output

### Tricky x axis

An issue that took way too long to fix was the x axis. When the DFT is calculated, the x values need to be remapped to the frequency range in use, otherwise you have incoherent data or something. Knew I ad to remap the x axis to something but did not know how. The lecturer helped me out on this one. By remapping the samples 0-N/2 to 0-Fs we get Hz on the x axis. APA Stimming

### DFT conclusion

A Discrete Fourier Analysis program was written by translating a BASIC program to C. It was compared to Excel’s build-in FFT function, Matlab and an online tool. It performs just as well as the platforms is was compared to.

## FSB

20Sim is a professional software package that was used during my previous education. The license for version 4.8 was obtained by my previous education. It runs Matlab under the GUI.

This paragraph details how the C code was iteratively generated. A feature is explained by way of 20Sim to then be implemented in C code. The resulting output is compared.

### The plant

Figure 11 shows the base 20Sim model. The RobotArm block is the single block representation of the block diagram shown in Figure 2. It has 3 outputs:

1. Position of the robotarm, in degrees
2. Speed of the robotarm, radians/second
3. Current drawn by the motor arm, in Ampère

Linearizing the model results in equation 1, the transfer function. Equation 2 is the factored version of equation 1. Figure 12 shows the Root Locus plot of the model. The poles are roughly at 0, -85 and -200.

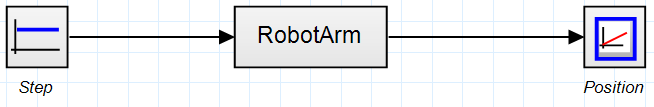


Figure 11 20Sim model

|  |  |
| --- | --- |
|  | (1) |
|  | (2) |



Figure 12 Root Locus plot plant

### Plant without FSB in C

One of the assignments during my previous education was to implement a second order transfer function on an arduino. The base code was written by one of the teachers (Moes, 2015). This code was extended to work with third order transfer functions, like in equations 1 and 2.

|  |  |
| --- | --- |
|  | (3) |
|  | (4) |
|  | (5) |
|  | (6) |

The transfer function is first rewritten into a state-space representation.

Example: equation 3 is broken up into equations 4, 5 and 6. These functions are converted to c code. The input is u and, in this case, is the output. The output is subtracted from the input at the beginning of each loop. A high-level overview of the code is given in Figure 13.

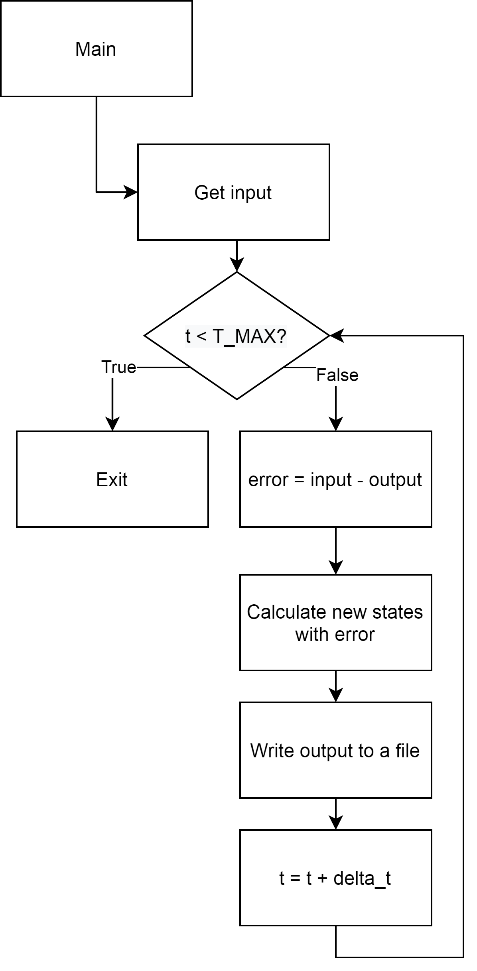


Figure 13 High Level overview fsb.c

Unit feedback was added to the model in Figure 11 in 20Sim and a step response was generated .

The output of fsb.c for a unit step response and unit is given Figure 14 and the output of 20Sim in Figure 15

Both give the same output.

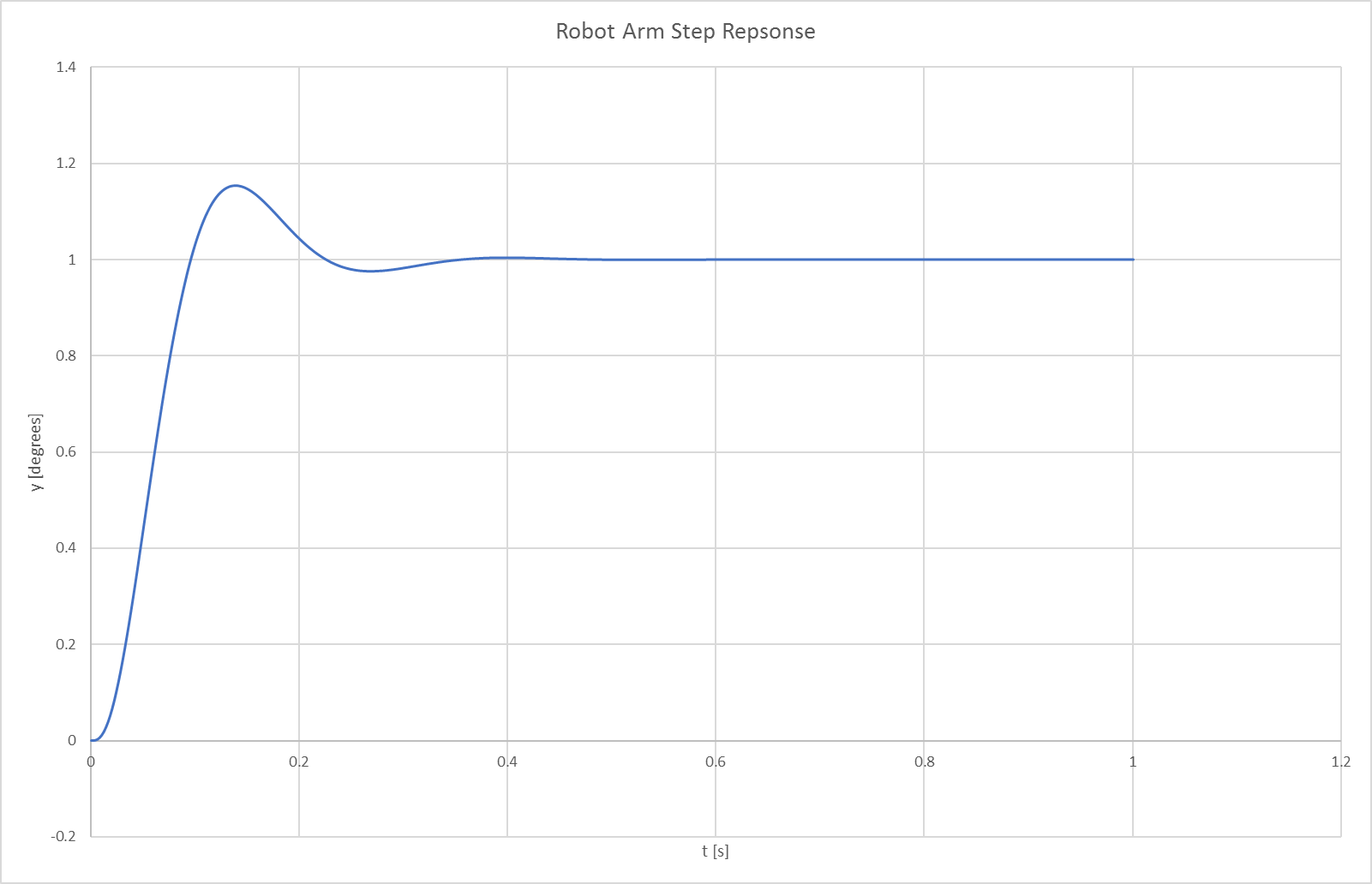


Figure 14 C code output on Step Response

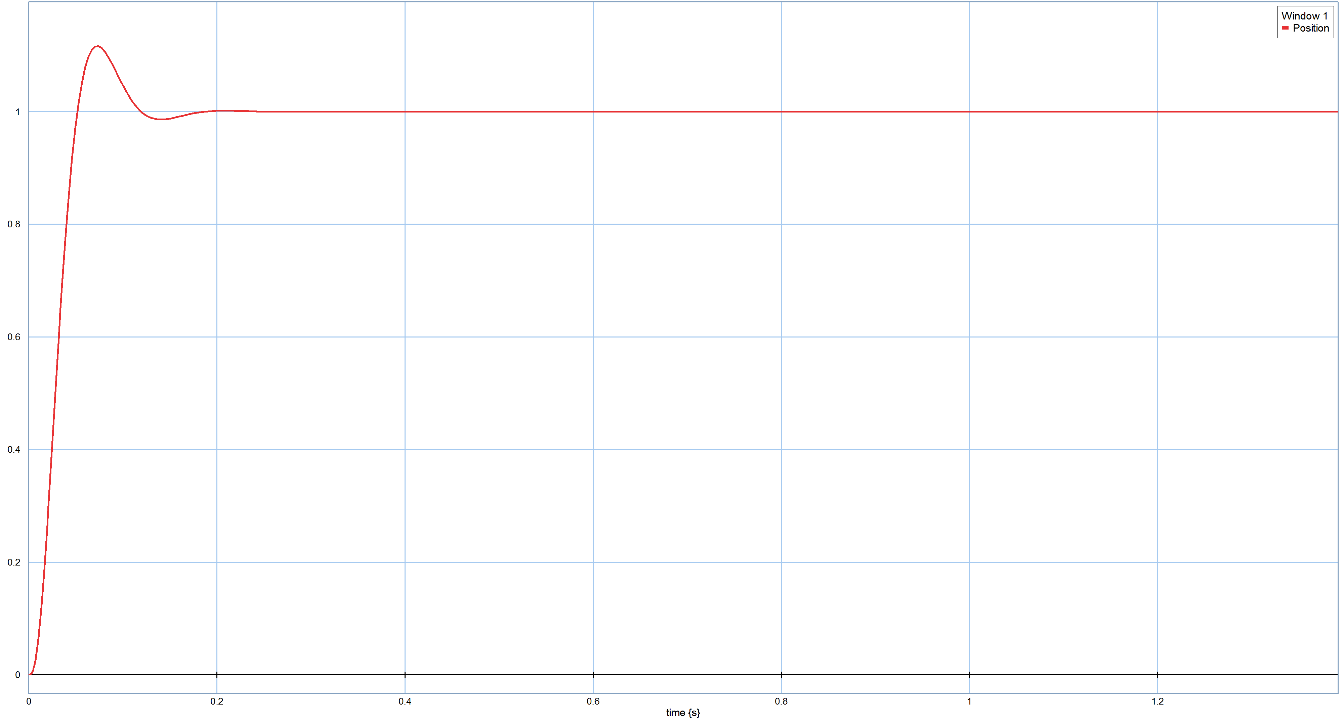


Figure 15 20Sim unit step response output

### Plant with FSB

20sim model with FSB

Resulting PN image

Step response comparison

As mentioned before, the Robot Arm model was used in a previous assignment. We implemented Full-State Feedback Control on the same plant. The resulting 20sim model is given in Figure 16. The blocks K1, K2 and K3 connect to the internal states of the RobotArm model. Their values were calculated using an Excel sheet provided by the teachers. Given a matrix describing a system and the desired pole locations, it uses the Ackerman formulas to calculate the corresponding K values (Hamberg & Moes, 2014).

The K values could alternatively be calculated by hand, as we did during the lectures.



Figure 16 20Sim RobotArm model with Full-State Feedback Control

Placing all three poles at -400 results in the step response shown in Figure 17. The system reaches steady state in 5ms instead of 600ms.

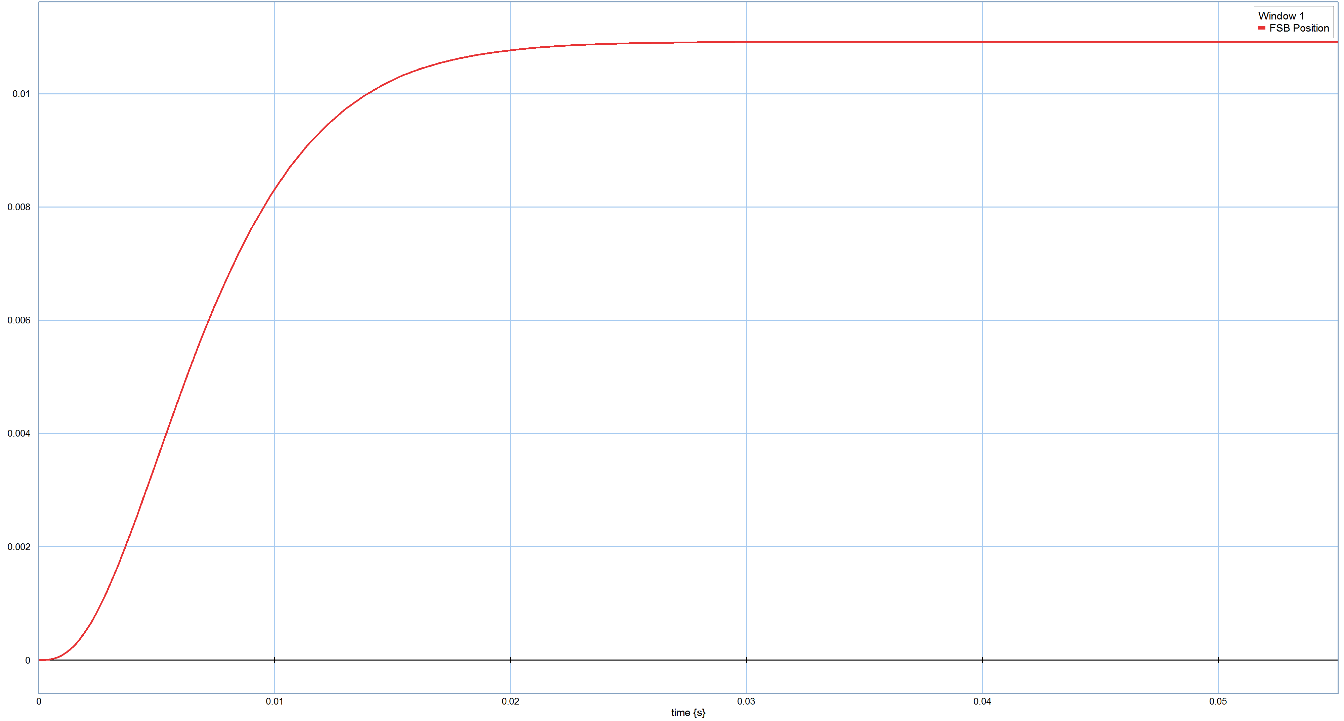


Figure 17 20Sim step response Robot Arm with Full-State Feedback control

### FSB in C

The problem with k values, discuss why it should work and how you determined that

Solution: new plant instead of k values

Comparison with 20sim

To implement FSB in C, two possible methods we tried.

1. Take the base plant, multiple states with K values each loop
2. Calculate new plant with FSB in 20Sim and implement that instead

Originally, method 1 would be used, because of personal preference. By plotting the outputs of the states (both in 20Sim and in C) the variables corresponding to the states were identified.

K3 is connected to x1, k2 to x2 and k1 to x3. Adding K2 and K3 to the C program worked fine. The output matched the 20sim simulation. Sadly, when adding K1 to the C program, the output went to infinity. The only reason I can think of is because the plant is a continues time plant, and the K values only work with discrete time plants, as we saw during the lectures.

The k values are:…

So I moved on the method 2. Linearizing the model given in Figure 16 results in equation 7. The step response of the C program is given in Figure 18, note that in the beginning the graph dips. This has to do with how Excel graphs smooth lines.

|  |  |
| --- | --- |
|  | (7) |

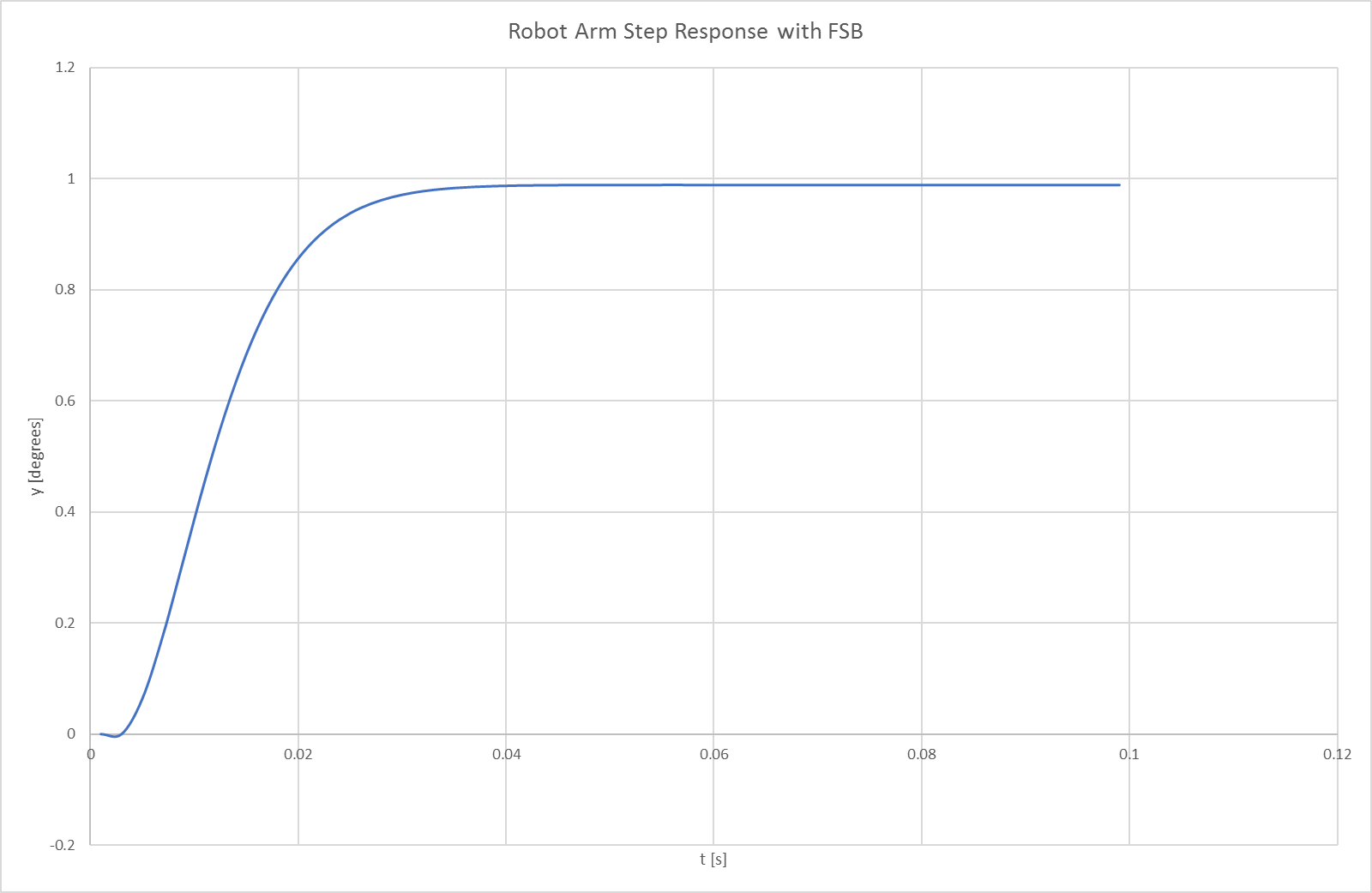


Figure 18 C code output on step response with FSB

### FSB Conclusion

A C program was written that reads an input value, runs it through a plant at writes the output to a text file. 20Sim was used to calculate the transfer function of the robot arm plant where the poles were placed using Ackerman theory.

20Sim was also used to validate the output of the C program.

## Visualization

The end goal of the visualizer is summed up in Figure 19. There would be a window divided into 4 quarters, each with its own function.

* Q1 is used to get input from the user such as sample frequency, sine frequency and phase.
* Q2 plots one period of the sine wave the user desires, super imposed with the sampled version of it.
* Q3 calls the DFT program and plots its output.
* Q4 calls the FSB program and plots its output.

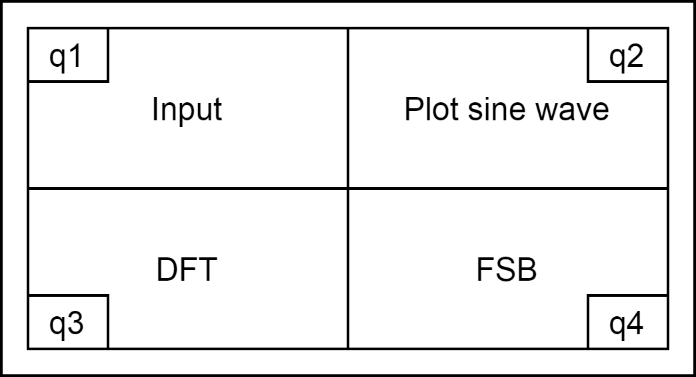


Figure 19 Template of the visualizer

An unwritten but possible use case for this project was that it could be used for educational purposes for my previous education and for the TUE. Therefor it would maybe need to run on different computers. It was decided to have the visualizer be an executable file, for Windows, that did not rely on other programs installed on the computer.

This paragraph details what choices were made during development of the visualizer.

### Choice of backend

My background is not in Graphical Design or Human Machine Interfaces. Therefor there were only two tools I was comfortable using:

* Python
* Processing

Processing was used during my previous education to make simple graphical user interfaces for our embedded projects. Even though programs have to be written in Java, there is a build-in function to export the project to an executable file. Sadly, the graphs in Processing did not look as good as the graphs generated by Matplotlib in Python, so I initially made the move to Python.

Python has a module called Tkinter which can be used to create GUIs (TkInter, 2020). It was used to create the input field widow in Figure 20. The graph in Figure 20 was made using Matplotlib (Matplotlib, 2020).

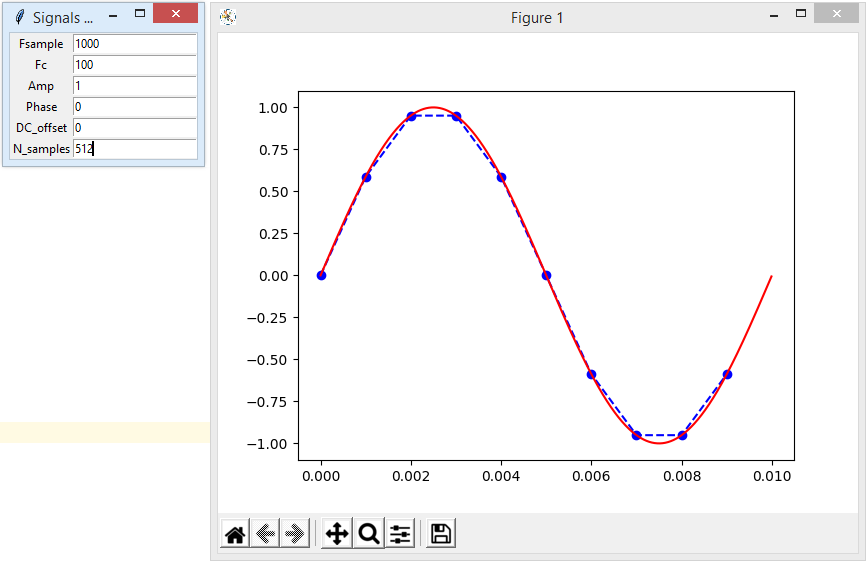


Figure 20: Python with Input and Graph windows

To convert python code to an executable, pyinstaller was used. It is run in the command line and generates its output in a specified directory (PyInstaller Manual, 2020).

The reason that Figure 20 does not include the DFT and FSB visualizers is because the python program would need to execute the executables of the corresponding programs. The issue is, after generating the executable of the python program Windows Defender kicks in and flags it as a Trojan Virus, as shown in Figure 21. This happens on both Windows 8.1 and Windows 10.

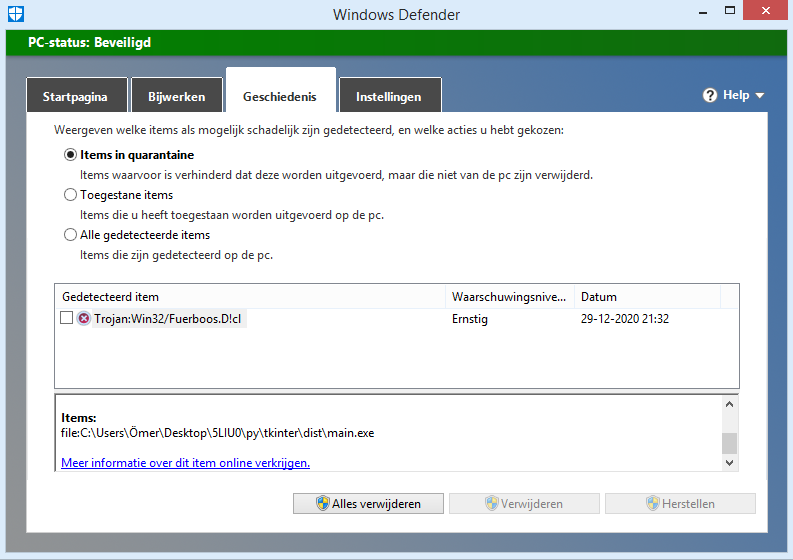


Figure 21 Windows Defender blocking the executable from running

Rewriting the programs into python would defeat the purpose as they were meant to run on embedded systems in C code.

There was an option to import the c programs as a module directly into python. Figuring out how took too much time and I moved on to Processing. In the next paragraph the final result of the processing implementation will be discussed.

### Final visualizer

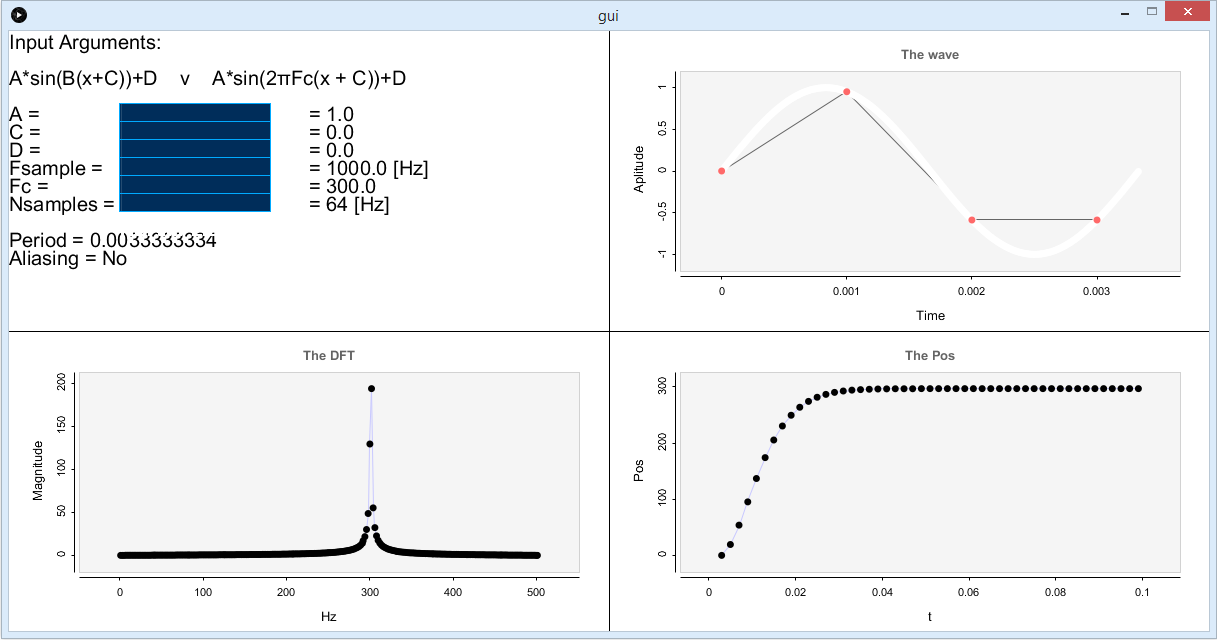


Figure 22 Final Visualizer

The template shown in Figure 19 is implemented in Processing and the result is given in Figure 22.

# Resultaten

[De opdrachtgever had een wens, die resulteerde in een opdracht. Er is mogelijk e.e.a. onderzocht, geanalyseerd, ontworpen, onderzocht, gebouwd, getest, geld uitgegeven en tijd besteed, en uiteindelijk een eindproduct gerealiseerd. De vraag is: wat valt er over dat eindproduct te zeggen?]

## Eindproduct

### Kenmerken

[Beschrijf hier o.a. aan de hand van de (meet- of onderzoeks)resultaten:

* eisen. Toon de eisen (bv. in tabelvorm; met dezelfde nummering en benaming als in eerdere hoofdstukken), en onderbouw in hoeverre deze gerealiseerd is.
* kenmerken. Wat het product kan.
* specificaties.

Alleen de feiten spelen hier een rol; in de evaluatie is ruimte voor meningen.]

### Evaluatie eindproduct

[Hier wordt objectief beschreven in hoeverre het product voldoet aan de specificaties? In hoeverre is het onderzoek valide? In hoeverre komt dit product overeen met de oorspronkelijke wensen van de opdrachtgever? Hoe ‘goed’ is het product? In hoeverre is het bv. gebruikersvriendelijk, of uitbreidbaar, platformonafhankelijk, modulair opgebouwd, veilig, stabiel, robuust, compact, snel, etc.

Verder, wat zijn zwakke punten in ontwerp, hard- of software, gebruikersvriendelijkheid?]

## Antwoord op de deelvragen

[Is er een probleemstelling geformuleerd met hoofd- en deelvragen, dan geeft deze paragraaf een antwoord op de deelvragen. Reserveer per deelvraag een subparagraaf.]

# Conclusie en aanbevelingen

## Antwoord op de hoofdvraag

[Is er een probleemstelling geformuleerd met hoofd- en deelvragen, dan geeft het hoofdstuk Conclusie een antwoord op de *hoofdvraag*. De conclusie is een interpretatie van het beeld zoals dat uit het hoofdstuk Resultaten naar voren komt. De feitelijke gegevens worden dan ook niet (of slechts globaal) vermeld, alleen de waardering die hieraan wordt gegeven. Uit het beschreven resultaat en evaluatie kan nu een conclusie getrokken worden, een eindoordeel. De Conclusie moet dus volgen uit voorgaande hoofdstukken en kan geen nieuwe informatie bevatten. De Conclusie moet, net als de Samenvatting, zelfstandig leesbaar zijn. Houd het kernachtig en overzichtelijk, bv. door elke conclusie in een aparte alinea te zetten of door nummering. ]

## Aanbevelingen

[De beschreven resultaten, evaluaties en conclusies kunnen leiden tot verschillende aanbevelingen aan de opdrachtgever, bv. over verbeterpunten van dit product of vervolgstappen. Het gaat daarbij om concrete, praktisch uitvoerbare handelingen.]

# Afkortingen en begrippen

[Elk verslag kan een lijst van illustraties, afkortingen, een verklarende woordenlijst en/of een register bevatten. Dit zijn gescheiden lijsten. De lijsten illustraties en afkortingen/symbolen kunnen ook voor in het verslag geplaatst worden, direct ná de samenvatting zonder hoofdstuknummer.

In plaats van deze lijsten kan gebruik gemaakt worden van voetnoten indien het aantal afkortingen en begrippen samen relatief gering is, bv. kleiner is dan tien. Let er op dat de voetnoot *altijd* geplaatst wordt bij de *eerste* keer dat de afkorting of het begrip voorkomt!]

## Lijst van Illustraties

[Lijst van illustraties, grafieken en tabellen. Rankschik deze op basis van volgorde. Geef aan: figuurnummer, titel en bladzijdenummer en een korte omschrijving.]

## Gebruikte afkortingen en symbolen

[Lijst van afkortingen, alfabetisch gerangschikt (eerst de Latijnse, dan de Griekse. Deze is nodig als er veel voorkomen in het verslag en als ze meer dan 1x gebruikt worden.]

## Verklarende woordenlijst

[Lijst van begrippen, alfabetisch gerangschikt, met dezelfde opmaak als de lijst van afkortingen. Deze is handig als een tekst voor meer doelgroepen met uiteenlopende achtergrondkennis geschikt moet zijn. Lezers die een term niet kennen, bladeren door naar de woordenlijst waar ze een duidelijke uitleg kunnen vinden. Zorg dat de uitleg bij een begrip zelfstandig te begrijpen is, verwijs zo min mogelijk naar andere begrippen op de lijst. Stem de uitleg af op dat deel van je doelgroep met de minste voorkennis.]

## Index

[De index bevat een alfabetische trefwoordenlijst met paginanummers. Dit is handig voor lezers die bepaalde informatie zoeken of voor lezers die alles willen bekijken wat over één onderwerp in de tekst wordt opgemerkt.]

# Bronnen

[Elk verslag heeft een *bronvermelding* of literatuurlijst. **Wil er geen sprake zijn van plagiaat, dan dienen *alle* geraadpleegde bronnen *altijd* vermeld te worden**. Neem alleen bronnen op waarnaar in het verslag verwezen wordt, dus geen extra bronnen. Er zijn verschillende systemen voor de vormgeving van de bronnen en verwijzingen. Binnen IED is gekozen voor de APA-standaard (American Psychological Association).

Wordt in de tekst gebruik gemaakt van citaten (letterlijke weergave) of aanhalingen/parafrases (in eigen weergeven wat een ander vermeld heeft) of plaatjes van internet, dan moet de bron altijd duidelijk zijn. Tevens dient bij een citaat *altijd* zichtbaar te zijn dat het een citaat is, bv. door gebruik van *cursivering* of "aanhalingstekens". Betreft het een aanhaling/parafrase, dan wordt de bron expliciet benoemd (bv. Elling zegt in Rapportagetechniek dat een ontwerprapport het ontwerp beschrijft van een apparaat (…). Zowel aan een citaat of parafrase wordt een voetnoot toegevoegd.

De *voetnoot* is zowel geschikt voor korte opmerkingen als ook voor bronverwijzingen. In hoofdstuk *bronvermelding* staat de bron in officiële APA-standaard vermeld. Hieronder een voorbeeld:

In de tekst:

Over de toetsing wordt gezegd: "*De opdrachten worden individueel getoetst.*" **1** Indien de toets niet gehaald wordt kan herkanst worden: "*De herkansing vindt plaats in de eerstvolgende toetsperiode.*"**2**

Voetnoten onder aan dezelfde pagina:

1. Jansen; *Leswijzer XYZ*; blz. 12
2. Jansen; *Leswijzer XYZ*; blz. 16

In hoofdstuk Bronnen:

Jansen (2014); *Leswijzer XYZ*. Utrecht: HU-FNT

Zie verder de bijlage over de APA-standaard voor bronvermelding.]

Elling, R., Andeweg, B. e.a. (2011); *Rapportagetechniek; schrijven voor lezers met weinig tijd*. Noordhoff Uitgevers Groningen/Houten. ISBN 9789001794781.

# Bijlagen

[Elk verslag kan bijlagen bevatten.

Bijlagen hebben optioneel een nummer en *altijd* een titel. Het is niet toegestaan om losse teksten of schema's als bijlage in het verslag 'te dumpen'. Meerdere bijlagen kunnen een aparte inhoudsopgave hebben. Mits mogelijk loopt de bladzijdenummering van het verslag door in de bijlage(n).

Wanneer komt iets in een bijlage? Alles wat bij de hoofdtekst hoort, wat nodig is om de hoofdtekst te kunnen volgen, komt niet in de bijlage. Het verslag dient dus zonder de bijlagen te begrijpen zijn. Omdat de hoofdtekst een compacte, samenhangende tekst dient te zijn, is het noodzakelijk om bijvoorbeeld deelthema's of uitgebreide verantwoordingen buiten de hoofdtekst te laten. Deze informatie kan dan in de bijlage geplaatst worden. Denk aan (technische) teksten, berekeningen, code’s, schema's, tabellen, testresultaten, interviewleidraad, enquêteformulier. Een technisch rapport zonder bijlagen behoort dus tot de uitzonderingen.

Tips. Begin elke bijlage op een aparte pagina. Verwijs in het verslag minimaal één keer naar elke bijlage en plaats de bijlagen in de volgorde waarin ze in het verslag aan bod komen. Bijlagen moeten, net als de Samenvatting en de Conclusie, zelfstandig leesbaar zijn.]

## Bijlage I: Aanwijzingen voor de schrijver

Het schrijven van een goede tekst is gecompliceerd. Hieronder volgen een aantal regels en tips voor de schrijver.

1. **Een document is een eindproduct**. Technische studenten zijn vaak te gericht op een werkend technisch eindproduct en onderschatten het belang van bv. het projectverslag: de robot werkt, maar het verslag is onder de maat. Vaak is echter een technisch product een ‘black box’, aan de buitenkant is voor de docent niet te zien hoe slim die in elkaar zit. In het verslag wordt aangetoond hoe en waarom iets op de gekozen manier gebouwd is. Het verslag laat zien hoe professioneel de student is. Daarom telt een verslag vaak zwaarwegend mee in de beoordeling.
2. **Schrijfproces**. Ga er vanuit dat het schrijven van de conceptversie van een tekst gemiddeld 20% uitmaakt van de totale tijd. Verreweg de meeste tijd, 80% dus, zit in het herschrijven en de afwerking. Datgene, wat door de student ingeleverd wordt, is vaak niet meer dan een conceptversie.
3. **Structuur en logica**. Zorg in het verslag voor een goede structuur. Zorg dat de opbouw van het verslag als geheel logisch in elkaar zit, en breng die logica ook aan in de aparte hoofdstukken en alinea's. Vaak zijn er meerdere mogelijkheden, het vergt denkwerk om de beste te kiezen.
4. **Argumenteren en verantwoorden**. De argumentatie toont het denkniveau aan. Het gaat er niet zozeer om wat gedaan is, als wel op welke gronden een keuze gemaakt is. Denk aan volgende zaken:
   1. worden stellingen en aannames altijd verantwoord?
   2. wordt duidelijk onderscheid gemaakt tussen waarneming en interpretatie?
   3. zijn de argumentaties logisch en voldoende onderbouwd?
   4. worden schema's en tabellen goed gebruikt (hebben ze een toegevoegde waarde) en verklaard?
   5. wordt de lezer meegenomen in het gebruik van formules en code-snippets?
   6. advies: wees alleen stellig in formuleringen als je dit ook onderbouwt of onderbouwd hebt! Gebruik anders termen als 'mogelijk', 'waarschijnlijk', 'aannemelijk'.
5. **Doelgroep**. Een tekst wordt in principe geschreven voor een ander. Bedenk dus goed: wat wil de lezer weten, wat vindt hij belangrijk, waar let hij op?   
   Realiseer je dat je met jouw tekst altijd een beroep doet op de lezer. Die moet tijd en energie vrijmaken om jouw tekst te doorgronden. Het is dus niet meer dan hoffelijk om het de lezer zo gemakkelijk mogelijk te maken. Met een slechte, slordige tekst dwing je de lezer extra energie te investeren om jouw tekst te begrijpen. Verplaats je dus in de lezer. Neem in de tekst de lezer stap voor stap mee in jouw gedachtengang. Ga er daarbij van uit dat de lezer hetzelfde opleidingsniveau heeft; leg dus geen zaken uit die 'onder' dat niveau liggen.
6. **Aanspreekvorm en taalgebruik**.
   1. Houd technische taal zakelijk. Gebruik daarvoor bij voorkeur de zg. passieve vorm (“geconstateerd werd…”) en niet de actieve vorm (“ik constateerde…”).
   2. Gebruik de ik- of wij-vorm alleen in voorwoord, evaluaties en reflecties. Verwijs niet naar jezelf in de derde persoon (niet: "De student heeft de opdracht gekregen om…", maar: "De opdracht luidt ….").
   3. De lezer wordt – in tegenstelling tot een presentatie – in een verslag niet direct aangesproken (vermijd: "u/je/jullie….").
   4. Wees concreet; gebruik geen onduidelijke begrippen als ‘redelijk’, ‘aardig’, ‘prima’, ‘leuk’, ‘fijn’ als het techniek betreft.
   5. Vermijd dat dezelfde tekst letterlijk meerdere keren terugkomt.
7. **Consequent**. Kies één stijl en één opmaak en houd daar aan vast. Wees consequent in:
   1. taalgebruik: hanteer steeds dezelfde stijlvormen, zoals de manier van formuleren, uitleggen, verwijzen.
   2. opmaak: zorg dat de pagina-opmaak steeds hetzelfde oogt.
   3. paragrafen en inspringen: zorg dat je steeds op dezelfde manier inspringt bij alinea’s en paragrafen.
8. **Spelling**. Vermijd spelfouten! Spelfouten kosten jou direct of indirect altijd ‘punten’. Zet dus altijd de spellingchecker aan, maar let op: ook de spellingchecker maakt fouten, vooral bij samenstellingen (zie onder). Bij twijfel, raadpleeg internet (bv. <http://woordenlijst.org>, [http://taalunieversum.org/spelling](http://taalunieversum.org/spelling/)). Veel gemaakte spelfouten zijn:
   1. samenstellingen. In het Nederlands bestaan in principe *geen woorden met spaties* er in. Je schrijft samengestelde woorden of aan elkaar, of je gebruikt een koppelteken (liggend streepje). Alleen bij Engelse (technische) begrippen mag je afwijken. Het is dus projectverslag, sensoringang (of sensor input), filtertechniek, cursusbeoordeling, meetresultaat. Zie ook <http://www.spatiegebruik.nl/>.
   2. d’s en t’s. Veel problemen leveren werkwoorden op die beginnen met ‘ver’ of ‘ge’, zoals vertellen en gebeuren. Het is “het gebeurt”, en “het is gebeurd”. Ook gaat er veel fout bij ’*t kofschip*.
   3. eenheden. Het is kΩ of kil*o*hm, en niet KΩ (kelvinohm), Mbyte en niet mbyte (millibyte)! Het is k [kilo], M [mega], G [giga], T [tera], Verder V [volt], Hz [hertz], J [joule], K [kelvin], W [watt], A [ampere], Ω [ohm], maar C [*C*elsius]. Bij twijfel, raadpleeg internet onder ‘SI’ (The International System of Units).
9. **Figuren, schema’s en plaatjes**. Wees consequent in de plaatsing van figuren, bv. steeds links uitgelijnd. Verder hebben figuren, schema’s en plaatjes *altijd* een titel, bv. “figuur 1.2”, en *altijd* dient naar een figuur vanuit de tekst verwezen te worden (“In figuur 1.2 wordt…” ). Van elk 'geleend' figuur dient de bron vermeld te worden, bv. m.bv. een voetnoot.
10. **Nummering**. Maak optimaal gebruik van bullets en nummering. Nummering vergroot de leesbaarheid. Gebruik bij opsommingen, bv. eisen, in principe altijd nummering. Vermijd 'talige' zinsconstructies als "ten eerste… ten tweede…, vervolgens…, ook….". Als een onderwerp uit meerdere zaken bestaat (bv. meerdere mogelijkheden, meerdere keuzes, meerdere stappen), pas dan nummering toe. Nummering heeft de voorkeur boven bullets, omdat daarnaar gemakkelijker verwezen kan worden.
11. **Tabellen**. Goed gemaakte tabellen vergroten de leesbaarheid en ogen professioneel. Met tabellen kun je makkelijk strakke en goed leesbare rijen en kolommen maken. Vergeet niet de kolombreedte aan te passen. Deze worden standaard vaak te breed ingesteld, waardoor de tabel onnodig breed wordt. Eventueel kun je de randen, de lijnen, onzichtbaar maken.
12. **Layout**. De layout (opmaak) is dienstbaar aan de tekst, de tekst wordt er beter leesbaar door. Tip. Bekijk een geprint vel van een meter afstand, en kijk of het goed, rustig, oogt. Aanbevolen maten zijn:
    1. lettertype: gebruik een zakelijk en goed leesbaar font (Calibri, Arial, Trebuchet MS)
    2. lettergrootte: gebruik voor de standaardtekst grootte 10 of 11.
    3. regelafstand: gebruik een regelafstand van minimaal 14pts. (Deze tekst heeft 15pts regelafstand.)
    4. kop- en/of voettekst: gebruik minimaal een van beide; met bv. naam, logo, paginanummer, totaal aantal pagina’s. Alleen een paginanummer onderaan volstaat niet.
    5. marges: gebruik marges rond de 2,5 cm. Houd bij de linkermarge rekening met de methode van inbinden!

## Bijlage II: De APA-standaard voor bronvermelding

Bronvermeldingen zijn gestandaardiseerd volgens de *APA-standaard (American Psychological Association):*

Boek:

Auteur, A.A., Auteur, B.B., & Auteur, C.C. (20xx). *Titel van het boek*. Plaats van uitgave: Uitgever. *Voorbeeld:*

Bouter, L.M., & Van Dongen, M.C.J.M. (1995). *Epidemiologisch onderzoek. Opzet en interpretatie*. Houten: Bohn Stafleu Van Loghum.

Hoofdstuk uit boek:

Auteur, A.A., Auteur, B.B., Auteur, C.C., & Auteur, D.D. (20xx). Titel van het hoofdstuk. In A. Editeur, B. Editeur, & C. Editeur (Eds.), *Titel van het boek* (pp. Xxx-xxx). Plaats van uitgave: Uitgever. *Voorbeeld:*

Meertens, R. W. (1992). Beïnvloedingsprocessen in groepen. In R. W. Meertens & Van Grumbkow (Eds.), *Sociale psychologie* (pp. 281-298). Groningen: Wolters-Noordhoff.

Tijdschriftartikel:

Auteur, A. A., Auteur, B. B., & Auteur, C. C. (20xx). Titel van het artikel. *Titel van het Tijdschrift*, xx (=volumenummer), xxx-xxx. *Voorbeeld:*

Houweling, H., Heisterkamp, S. H., Van Wijngaarden, J.K., & Wiessing, L. G. (1994). Analyse van de AIDS-epidemie in Nederland van 1982 tot 1993. *Nederlands Tijdschrift voor Geneeskunde*, 138, 238-259.

CD-rom:

Bij een CD-rom wordt dezelfde methode gehanteerd als bij een boek. Echter krijgt het de toevoeging CD-rom aan het einde.

Online document:

Auteur, A.A., Auteur, B.B., & Auteur, C.C. (20xx). *Titel van het artikel*. Datum van opvragen gegevens (dd-mm-jaar), URL-adres. Is de auteur niet bekend, vul dan Anonymus in (let dan wel op de betrouwbaarheid van de gevonden informatie. *Voorbeeld:*

Steenhuis, P. (2001). Wordt Harry uiteindelijk baas van Zweinstein? 05-10-2001, http://www.nrc.nl/Potterhoekje/voorspelling.html.

Zie verder:

* <http://www.apastyle.org/elecmedia.html>;
* APA-documenten op de site *leerlijn Professionalisering*. ]

## Bijlage III: Plagiaat

In deze bijlage gaan we kort in op plagiaat[[1]](#footnote-1).

Alle verslagen worden met behulp van het antiplagiaatprogramma Ephorus gescreend!

Plagiaat is:

* het kopiëren van (gedeelten van) verslagen of ander werk van andere studenten;
* het kopiëren van informatie van internet of andere informatiebronnen zonder *correcte* bronvermelding.   
  Voor correcte bronvermelding: zie de bijlage over de APA-standaard.

Bij vermeende constatering van plagiaat vindt de volgende procedure plaats:

1. bij de examencommissie komt er een melding binnen;
2. de student wordt altijd opgeroepen voor een hoorzitting met 2 leden van de examencommissie;
3. er wordt een besluit genomen in de eerstvolgende vergadering van de examencommissie;
4. wanneer de onregelmatigheid vastgesteld wordt, volgt er een sanctie.

Mogelijke sancties zijn:

* altijd: opname in dossier van de student;
* een Niet Geldig (NG)-verklaring van de toets;
* een uitsluiting van alle toetsing (per blok, per semester of per jaar);
* een advies tot beëindiging van de opleiding aan het opleidingsmanagement.

Zie verder het *Onderwijs- en Examenreglement* (OER).

## Bijlage IV: Dit format en SCRUM

Ook als je het project met Scrum gerealiseerd hebt, neem je nog steeds het *format Verslag* als uitgangspunt. Let op de volgende hoofdstukken:

1. **Hoofdstuk [Project / Opdracht]**. Dit hoofdstuk blijft ongewijzigd.

Voor de eisen en randvoorwaarden ga je echter uit van de product backlog. Zie punt 3 als er geen product backlog is, maar sprake is van eisen en randvoorwaarden.

1. **Hoofdstukken Analyse, 5. Ontwerp**. Deze hoofdstukken blijven in principe ongewijzigd.

Bouw deze hoofdstukken niet chronologisch, maar logisch op. Belangrijk: maak inzichtelijk welke keuzes gemaakt zijn, en onderbouw dit.

1. **Hoofdstuk Proces en Planning**.

Beschrijf hier hoe de Scrum-methode toegepast is of gaat worden. Doe dit aan de hand van een korte toelichting op de verschillende Scrum-onderdelen. Dit is reeds beschreven in hoofdstuk Proces en Planning.

1. [nieuw/optioneel hoofdstuk] **Transitie naar Scrum**.

Indien de opdracht op de standaardwijze gedefinieerd is met eisen en randvoorwaarden is het handig om dit hoofdstuk in te voegen. Je kunt dan het opdracht/analyse/ontwerpdeel gewoon aanpakken op de watervalmanier. Op basis van de eisen en opdrachtanalyse maak je in dit hoofdstuk een vertaalslag naar de scrum-methode, zodat je de 'realisatiefase' met scrum kunt uitvoeren.

Daarvoor is nodig dat alle eisen, randvoorwaarden en wensen stuk voor stuk helder vertaald worden naar een product backlog met user stories. Dit hoofdstuk bevat dan een lijst met deze user stories. De user stories hebben:

* + een prioriteit (optioneel).
  + de juiste formulering, afgestemd op de verschillende rollen ("als opdrachtgever wil ik dat…", "als gebruiker wil ik…", "als student …" etc.). Dus ook die zaken, die jij als student moet opleveren (bv. verslagen of uit te voeren pops) worden als user story gedefinieerd. Advies: betrek de opdrachtgever, vanaf nu product owner, hierbij.
  + acceptance criteria (bv. DIN-normen, meetwaarden).
  + de Definition of Done.
  + de geschatte uren (m.b.v. scrum planning poker).

Nadat de PBL bekend is, kan de eerste sprint planning meeting gehouden worden (waarbij in deel 1 de product owner aanwezig is - lukt dit niet, dan moet het team duidelijk aangeven, hoe de product owner-verantwoordelijkheid door het team gewaarborgd wordt).

1. **Hoofdstuk Realisatie**.

Ga hier uit van de afzonderlijke Sprints, dus chronologisch. Beschrijf per Sprint: 1. doel, bouw (user stories & sprint tasks), eindproduct, test/demo, conclusie. De Sprint Reviews vormen de basis hiervoor.

1. **Hoofdstuk Eindproduct**.

Voeg na *Evaluatie* een paragraaf *Validatie* toe, waarbij gevalideerd wordt in hoeverre het eindproduct overeenkomt met de gestelde 'eisen' van de Product Backlog. Je zult hier de product owner in moeten betrekken. (Bij Evaluatie evalueer je de technische kwaliteit van het product).

1. **Hoofdstuk Zelfanalyse.**

Dit hoofdstuk vormt de samenvatting van de Sprint Retrospective meetings. De paragraafindeling wordt hierop aangepast:

*§1 Teamreflectie*. Beschrijf de sterke en verbeterpunten van het team.

*§2 Individuele prestaties*. Maak per teamlid een samenvatting van de verschillende peerassessments tot 1 tabel.

# Bibliography

Ahmed, A. (Ed.). (2013, December 13). *FFT calculator*. Retrieved January 29, 2021, from https://scistatcalc.blogspot.com/: https://scistatcalc.blogspot.com/2013/12/fft-calculator.html

Moes, A. (2015, October 3). Sysdyn\_tweede\_orde. Hogeschool Utrecht Institute for Engineering & Design.

Smith, S. W. (1999). *The Scientists and Engineer's Guide to Digital Signal Processing.* US: California Technical Publishing. Retrieved December 13, 2020, from http://www.dspguide.com/pdfbook.htm

1. bron: Bor, E.van de, Examencommissie Elektrotechniek; *Presentatie eerstejaarsstudenten 2013.ppt*; HU FNT, Utrecht 2013. [↑](#footnote-ref-1)